Algorithms: Solutions 2

```
               X
               X
               X
               X
               X
               X
               X     X
               X     X
               X     X
            X  X     X
            X  X     X
         X  X  X  X  X
   X  X  X  X  X  X  X
   X  X  X  X  X  X  X
X  X  X  X  X  X  X  X
X  X  X  X  X  X  X  X
-----------------------
3  4  5  6  7  8  9  10
          grades
```

The histogram shows the distribution of grades, from 0 to 10.

Problem 1
Give an example of functions $f(n)$ and $g(n)$ that satisfy all of the following conditions:

$f(n) = O(g(n))$
$f(n) \neq \Theta(g(n))$
$f(n) \neq o(g(n))$

Consider the following two functions:

$$\begin{aligned} f(n) &= 1 \\ g(n) &= \begin{cases} 1 & \text{if } n \text{ is even;} \\ n & \text{if } n \text{ is odd.} \end{cases} \end{aligned}$$

Since $f(n) \leq g(n)$, we immediately conclude that $f(n) = O(g(n))$. For even n, the function $f(n)$ is of the same order as $g(n)$, which means that $f(n) \neq o(g(n))$. On the other hand, for odd n, $f(n)$ grows asymptotically slower than $g(n)$, which implies that $f(n) \neq \Theta(g(n))$.

Problem 2
Give a precise mathematical proof of the following asymptotic bounds:

(a) $\sqrt{n} = o(n)$

We need to show that, for every $c > 0$, there is some n_0 such that, for all $n \geq n_0$, we have $\sqrt{n} < c \cdot n$. We define n_0 as follows:

$$n_0 = \left\lceil \frac{1}{c^2} + 1 \right\rceil.$$

Then, for every $n \geq n_0$, we have $n > 1/c^2$, which implies that $\sqrt{n} \cdot c > 1$ and readily leads to the desired inequality:

$$\sqrt{n} < \sqrt{n} \cdot (\sqrt{n} \cdot c) = n \cdot c.$$

(b) $(n+1)^a = \Theta(n^a)$

If $n \geq 1$, then

$$(n+1)^a \leq (2n)^a = 2^a \cdot n^a.$$

Thus, we get the following bounds for $(n+1)^a$:

$$n^a \leq (n+1)^a \leq 2^a \cdot n^a,$$

which implies that $(n+1)^a = \Theta(n^a)$.

Problem 3
Prove the following transitivity property of asymptotic bounds:

if $f(n) = \Theta(g(n))$ and $g(n) = \Theta(h(n))$, then $f(n) = \Theta(h(n))$.

Since $f(n) = \Theta(g(n))$, we conclude that there are some positive constants c_1, c_2, and n_1 such that, for all $n \geq n_1$, we have:

$$c_1 g(n) \leq f(n) \leq c_2 g(n).$$

Similarly, since $g(n) = \Theta(h(n))$, there exist some positive constants c_3, c_4, and n_2 such that, for all $n \geq n_2$:

$$c_3 h(n) \leq g(n) \leq c_4 h(n).$$

We may combine these two inequalities as follows:

$$c_1 c_3 h(n) \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \leq c_2 c_4 h(n).$$

We now define three new constants, c_5, c_6, and n_3:

$c_5 = c_1 c_3,$
$c_6 = c_2 c_4,$
$n_3 = \max(n_1, n_2).$

Then, the last inequality implies that, for every $n \geq n_3$, we have:

$$c_5 h(n) \leq f(n) \leq c_6 h(n).$$

This inequality means that, by definition, $f(n) = \Theta(h(n))$.

Problem 4

Suppose that we have four algorithms, called A_0, A_1, A_2, and A_3, whose respective running times are n, n^2, $\lg n$, and 2^n. If we use a certain old computer, then the maximal sizes of problems solvable in an hour by these algorithms are s_0, s_1, s_2, and s_3.

Suppose that we have replaced the old computer with a new one, which is k times faster. Now the maximal size of problems solvable in an hour by A_0 is $k \cdot s_0$. What are the maximal problem sizes for the other three algorithms, if we run them on the new computer?

For A_1: On the old machine, the A_1 algorithm solves a problem of size s_1 in one hour. The running time of this algorithm on a problem of size s_1 is s_1^2; hence, $s_1^2 = 1$ hour.

The new machine is k times faster, which means that the running time of A_1 is n^2/k. We denote the size of the largest problem solvable in one hour by v_1; then, $v_1^2/k = 1$ hour.

We conclude that $v_1^2/k = s_1^2$ and, hence, $v_1 = s_1\sqrt{k}$. Thus, the maximal size of a problem solvable in one hour on the new machine is $s_1\sqrt{k}$.

For A_2: On the old machine, the A_2 algorithm solves a problem of size s_2 in one hour, which means that $\lg s_2 = 1$ hour. If we denote the maximal problem solvable in an hour on a new machine by v_2, then $\lg v_2/k = 1$ hour. We conclude that $\lg v_2/k = \lg s_2$, which implies that $v_2 = s_2^k$. Thus, the maximal problem solvable in one hour on the new machine is of size s_2^k.

For A_3: We denote the maximal problem solvable by A_3 on the new machine by v_3, and use a similar reasoning to obtain the equation $2^{v_3}/k = 2^{s_3}$, which implies that $v_3 = s_3 + \lg k$.